DevOps Round-trip Software Engineering: On Traceability from Dev to Ops and back

Miguel Jiménez, Hausi A. Müller
{miguel, hausi}@uvic.ca

Norha Villegas, Gabriel Tamura
{nvillega, gtamura}@icesi.edu.co

Joe Wigglesworth, Ian Watts
{wiggles, ifwatts}@ca.ibm.com

May 10, 2018

London, ON, Canada

IN THE BEGINNING

DEVELOPMENT AND OPERATIONS WERE FUNCTIONAL SILOS

Let there be

DevOps

Integration

Culture

Communication

Delivery

Measurement

Quality

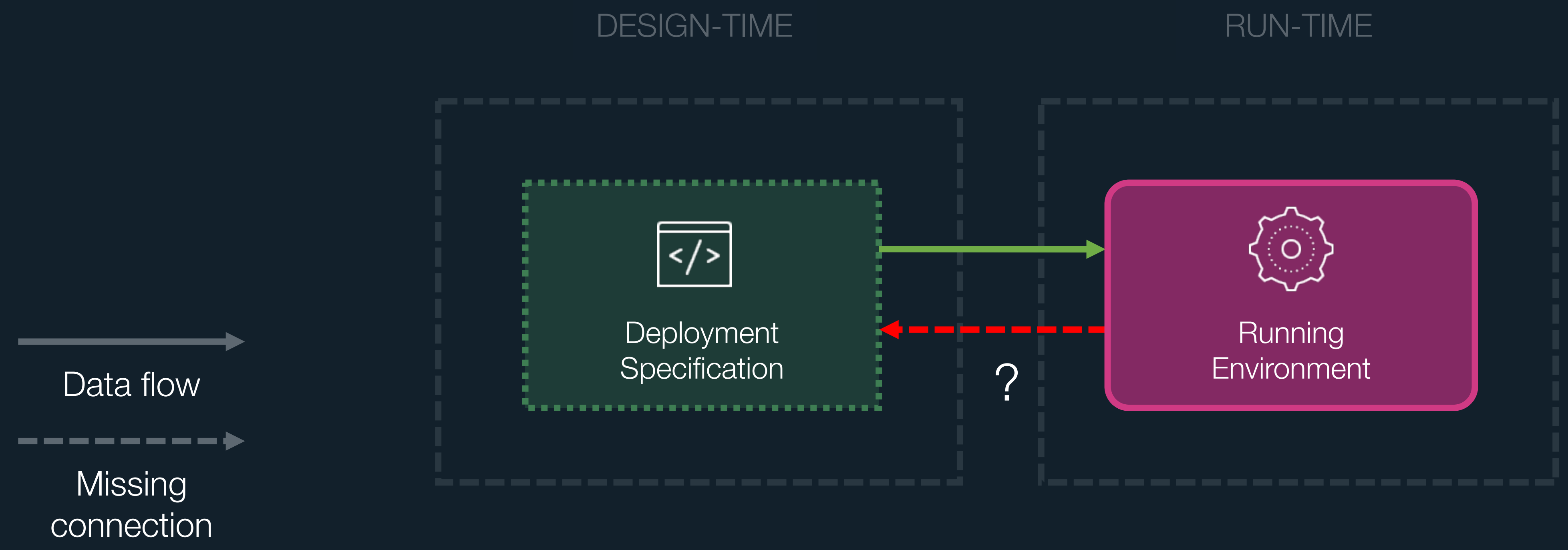
Automation

Tools

Improvement

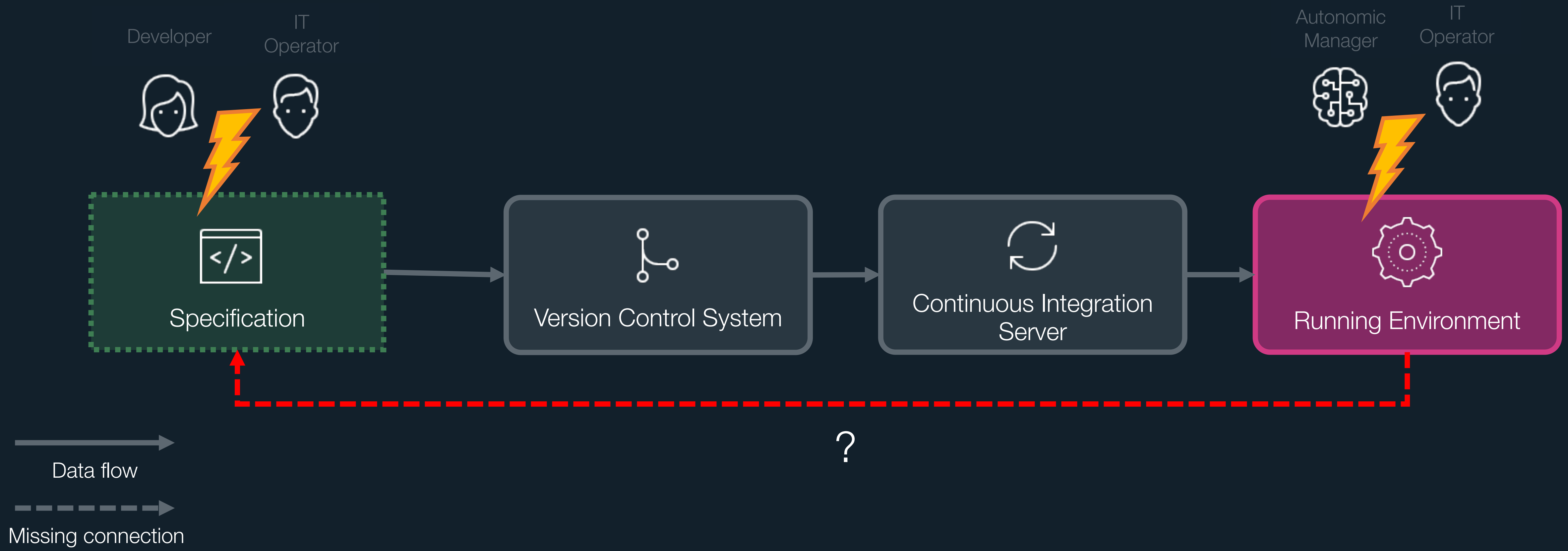
DevOps Continuous Cycle

- Changes at runtime don’t affect design-time specs
- Inconsistencies lead to configuration drift, snowflake configuration and erosion



Testing Deployment Specs

- Experimenting on testing environments enables IT operators to develop new features and fix faults by performing ad-hoc modifications
- Testing specifications requires deploying them (time + money)

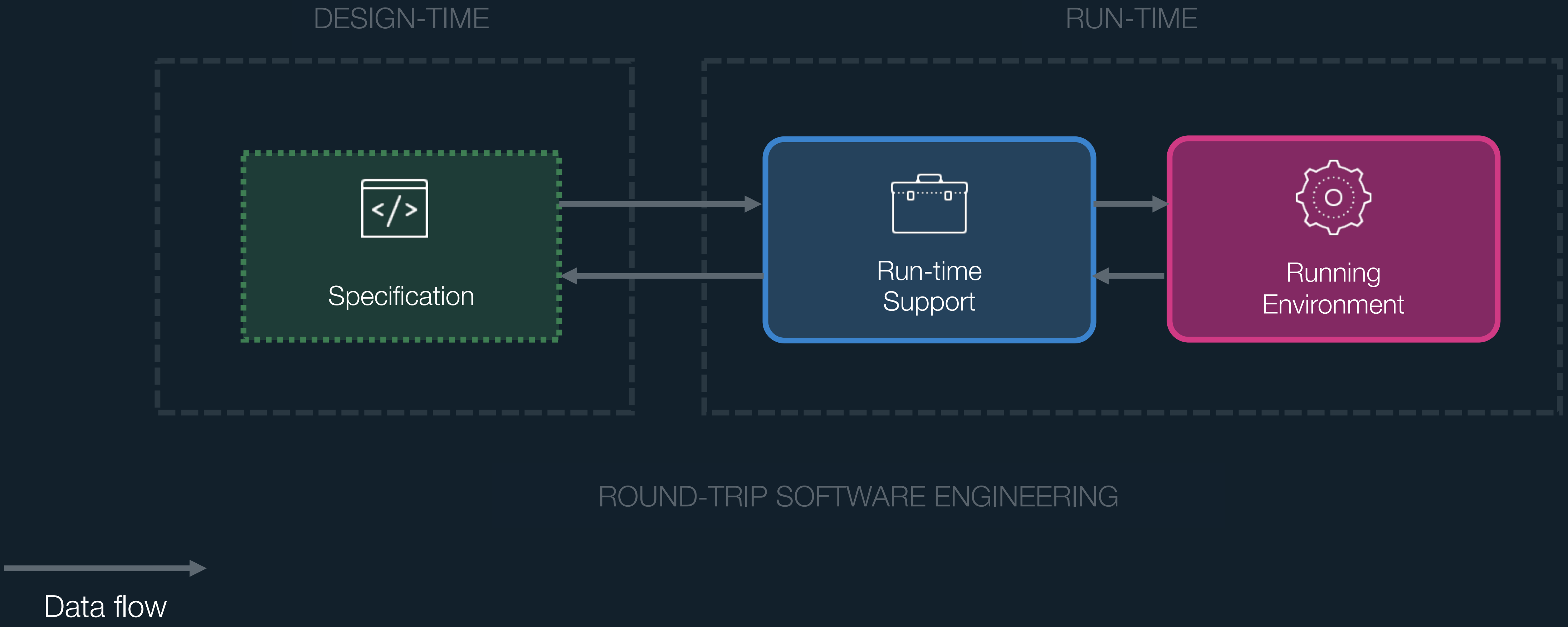


Framework Proposal

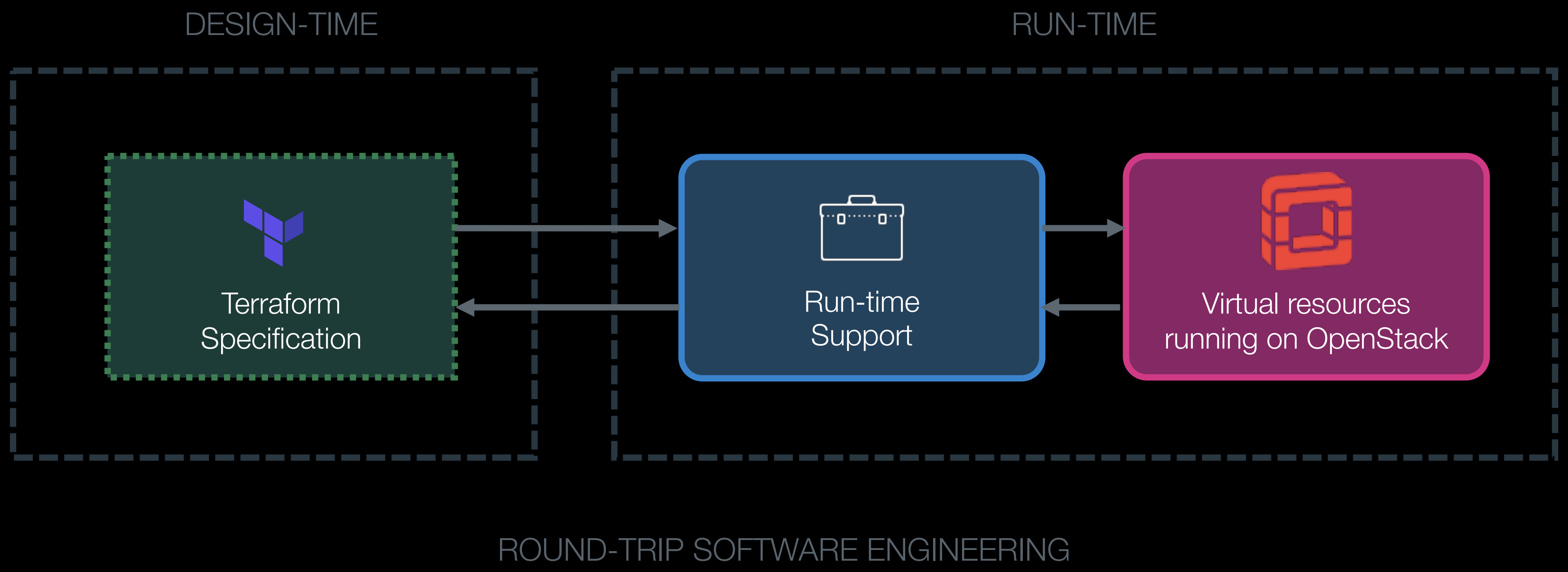
Goals:

1. To keep design-time specifications in sync with the running system
2. To integrate runtime data from operations back to development
3. To reduce testing time

Framework Overview



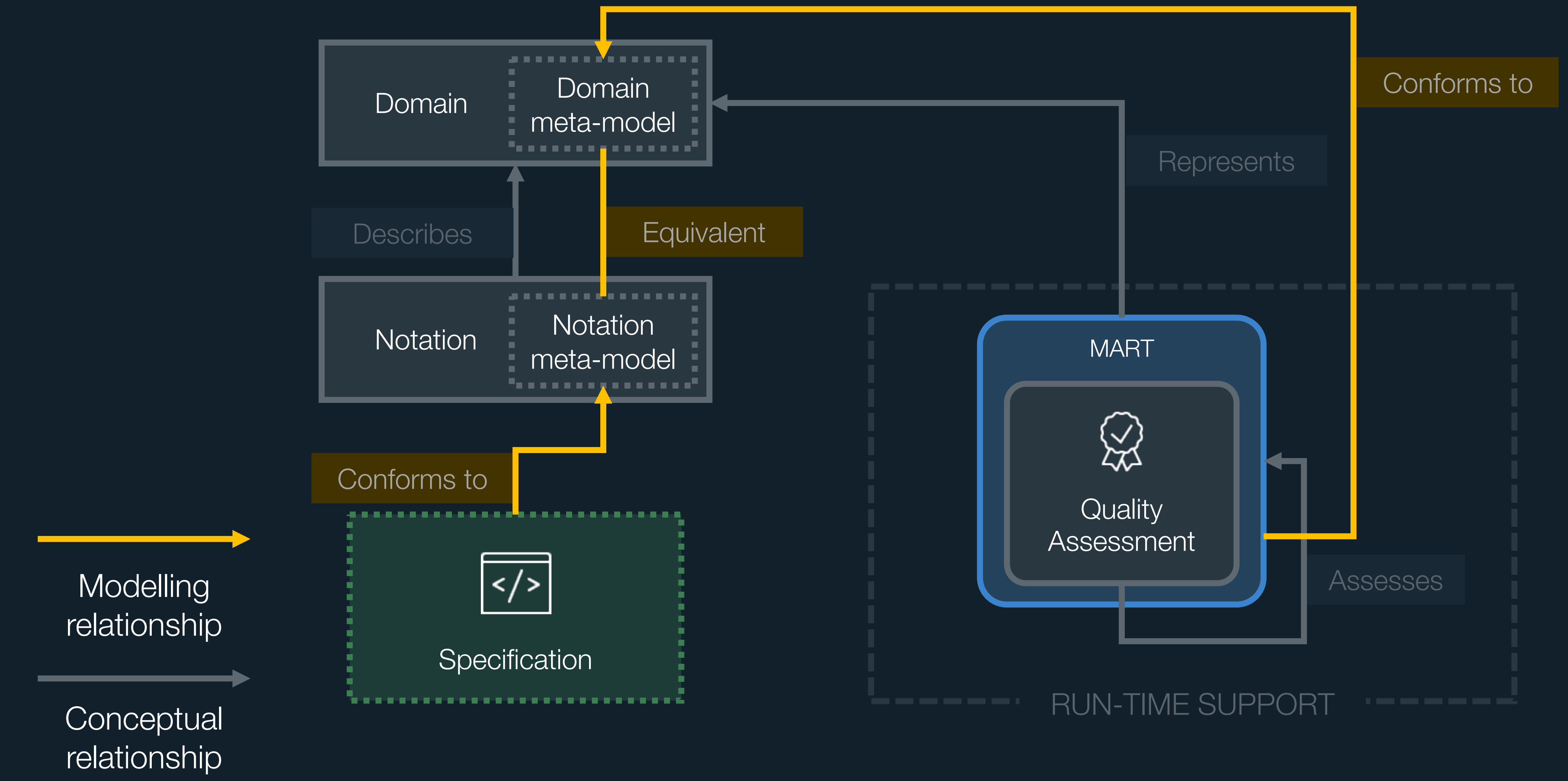
Running Example: Terraform and OpenStack


DESIGN-TIME
RUN-TIME
Terraform Specification
Run-time Support
Virtual resources running on OpenStack
ROUND-TRIP SOFTWARE ENGINEERING

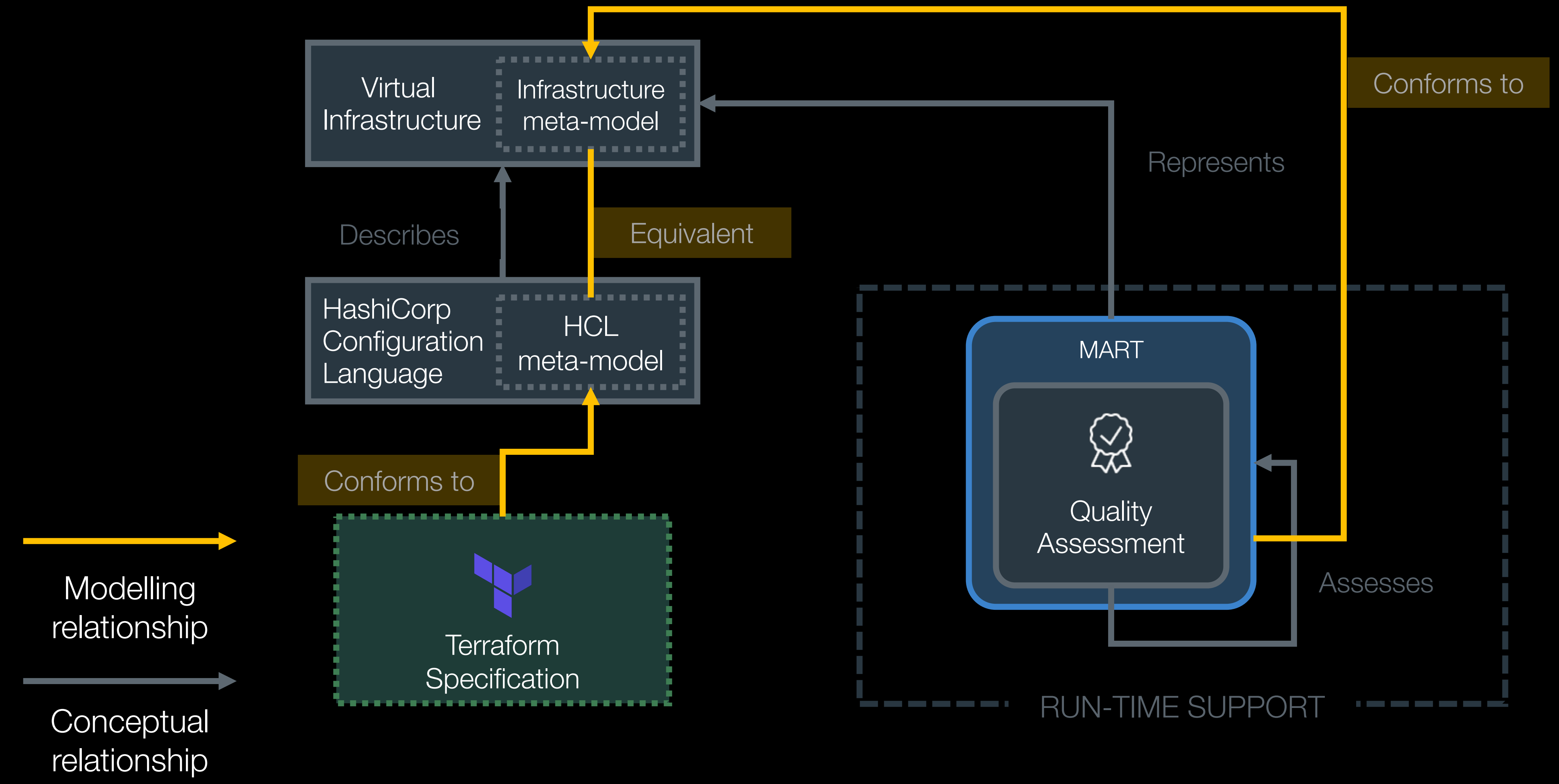
Running Example: Terraform Spec

```
variable "image" {
    default = "Ubuntu 14.04”
}
resource "openstack_compute_keypair_v2" "terraform" {
    name = "terraform”
    public_key = "${file("~/.ssh/id_rsa.terraform.pub")}”
}
…
output "address" {
    value = "${openstack_compute_floatingip_v2….address}”
}
```

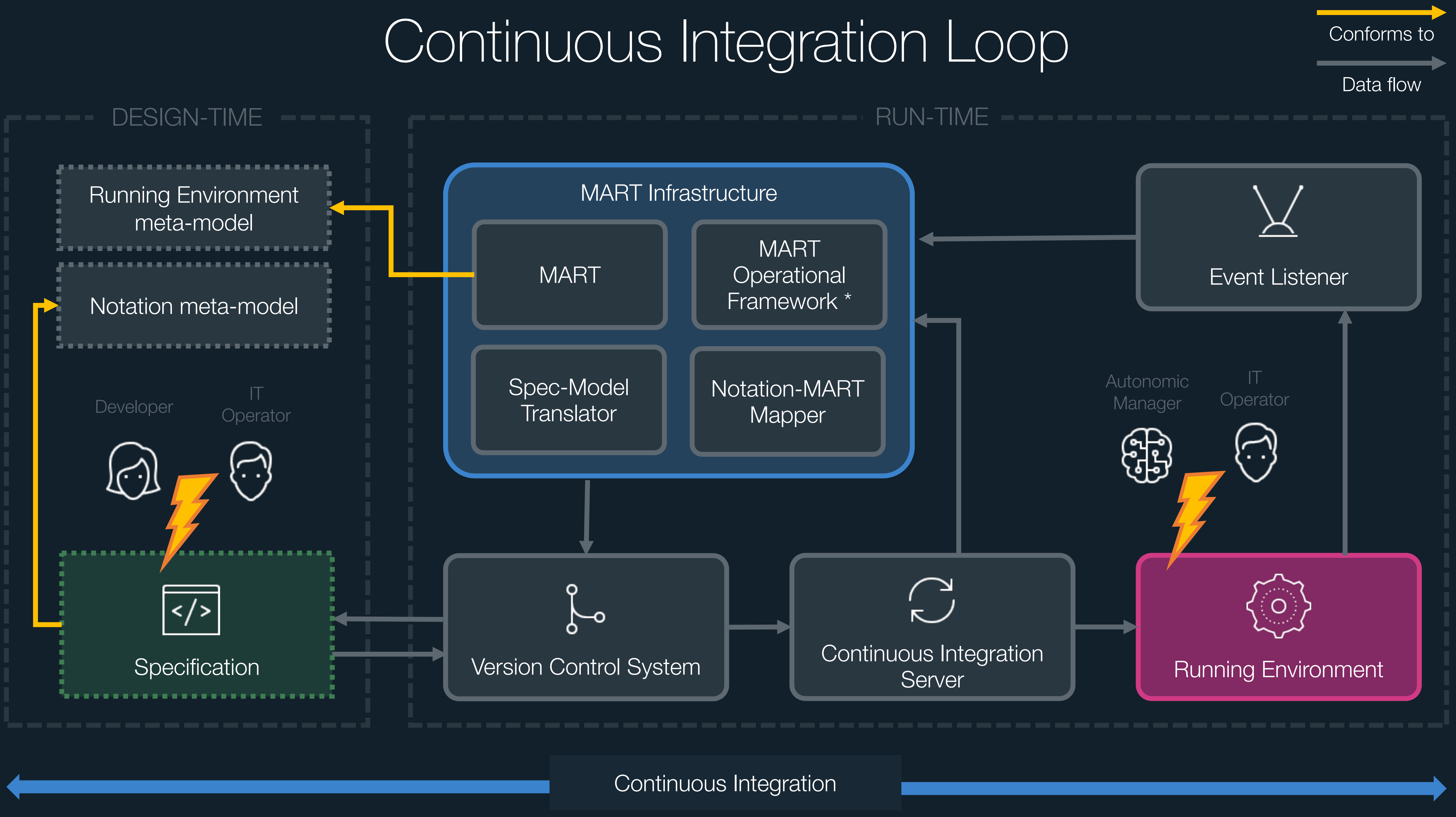
Relationship Between Notation and MART



Running Example: Terraform and OpenStack

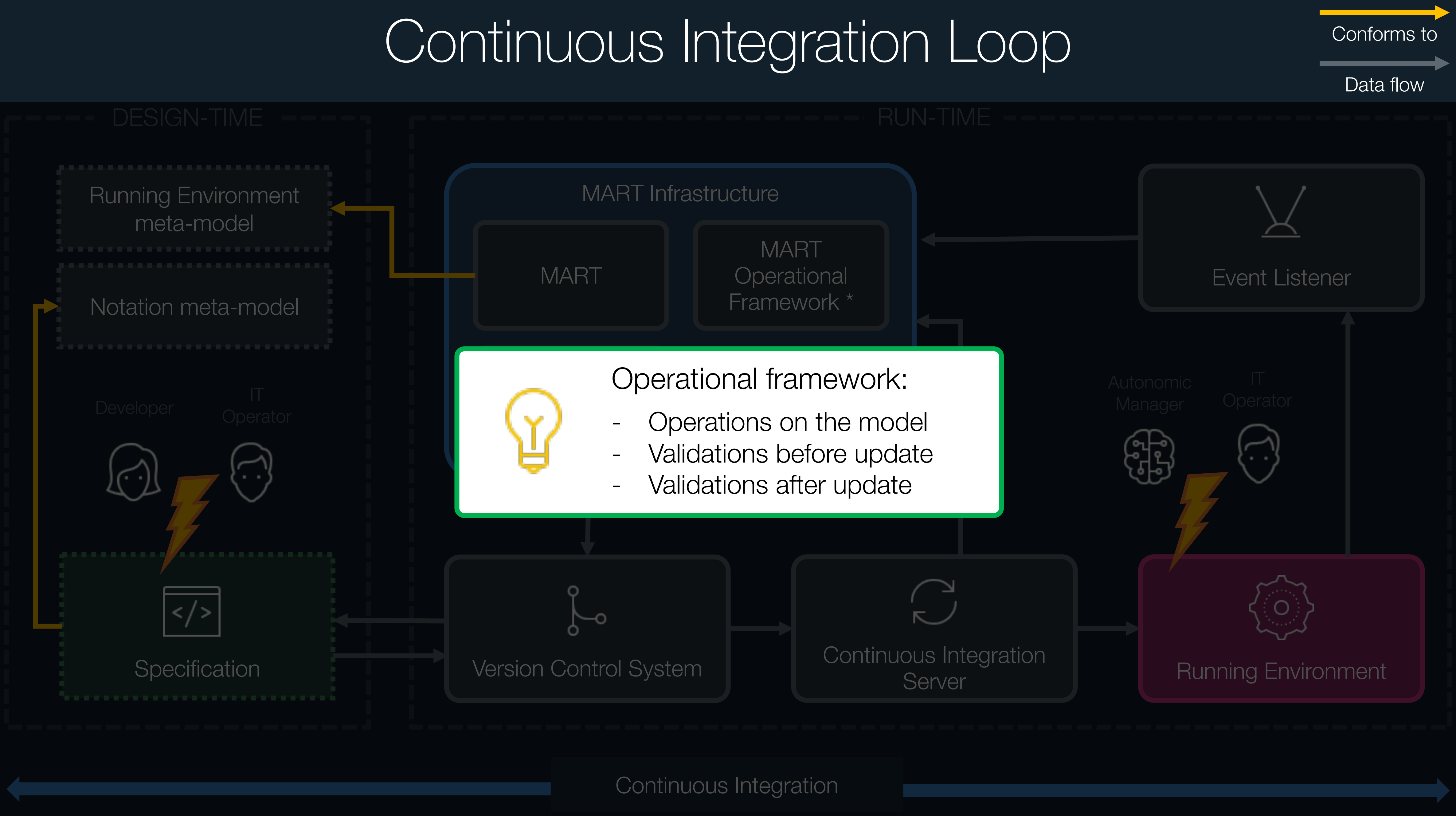


Continuous Integration Loop



* Castañeda, Lorena. "Runtime Modelling for Smart User-centric Cyber-Physical-Human Applications". PhD thesis, 2017

Continuous Integration Loop

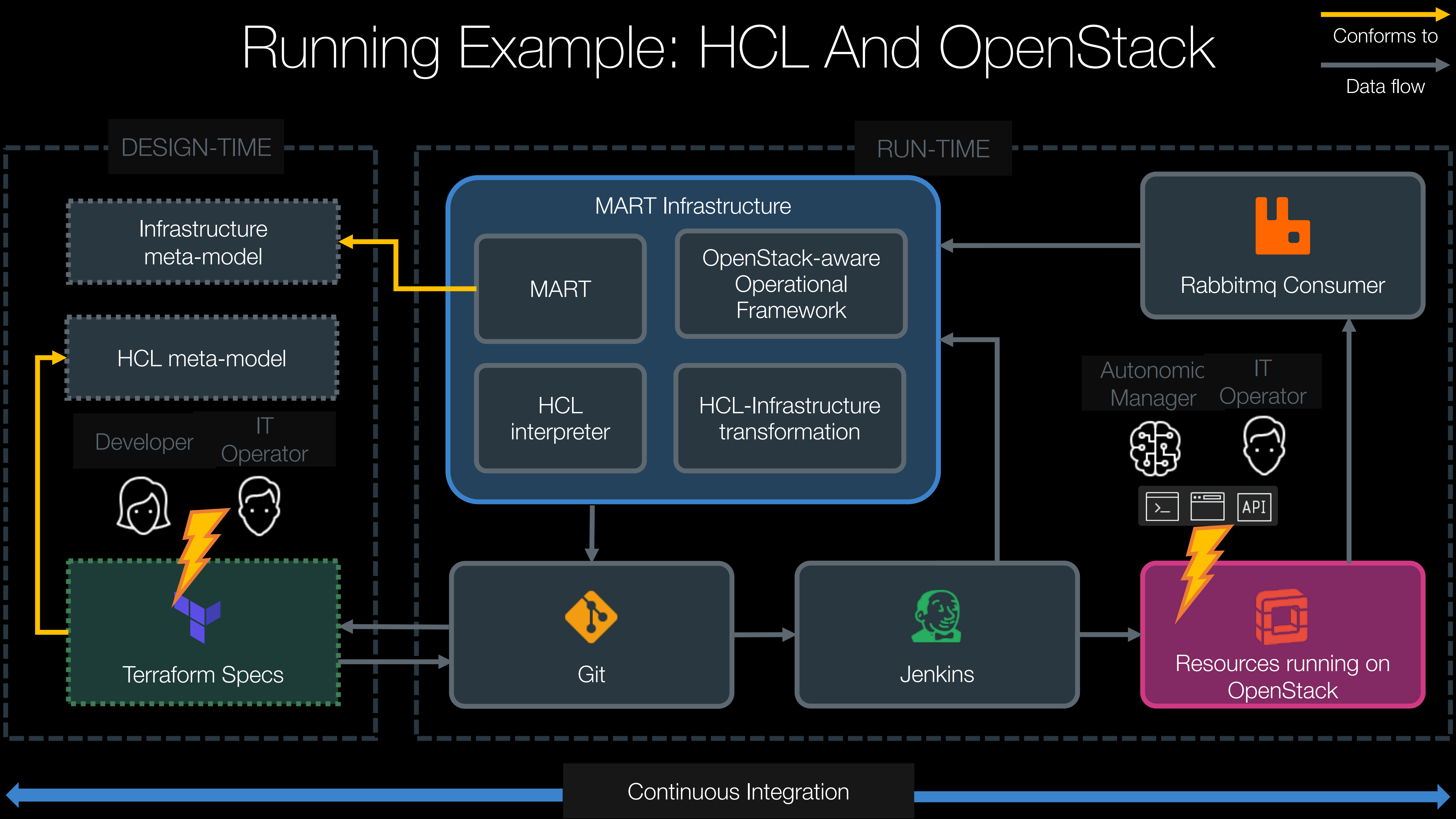


Operational framework:

- Operations on the model
- Validations before update
- Validations after update

* Castañeda, Lorena. "Runtime Modelling for Smart User-centric Cyber-Physical-Human Applications". PhD thesis, 2017

Running Example: HCL And OpenStack



Contribution Model: integration of changes

MART infrastructure as:

1. COMMITER

PROS

- No delay to reflect changes
- Less merge conflicts

CONS

- Risk. Unsupervised changes can break the build

2. CONTRIBUTOR

PROS

- No risk

CONS

- Delay to reflect changes
- Extra time spent reviewing changes
- Merge conflicts are expected
- Specs and MART are inconsistent while pull request remains open unless the MART goes back to the previous version

Contribution Model: integration of changes

MART infrastructure as:

1. COMMITER

PROS

- No delay to reflect cha
- Less merge conflicts

CONS

- Risk. Unsupervised changes can break the build

Certain types of changes are directly committed while others are requested

2. CONTRIBUTOR

PROS

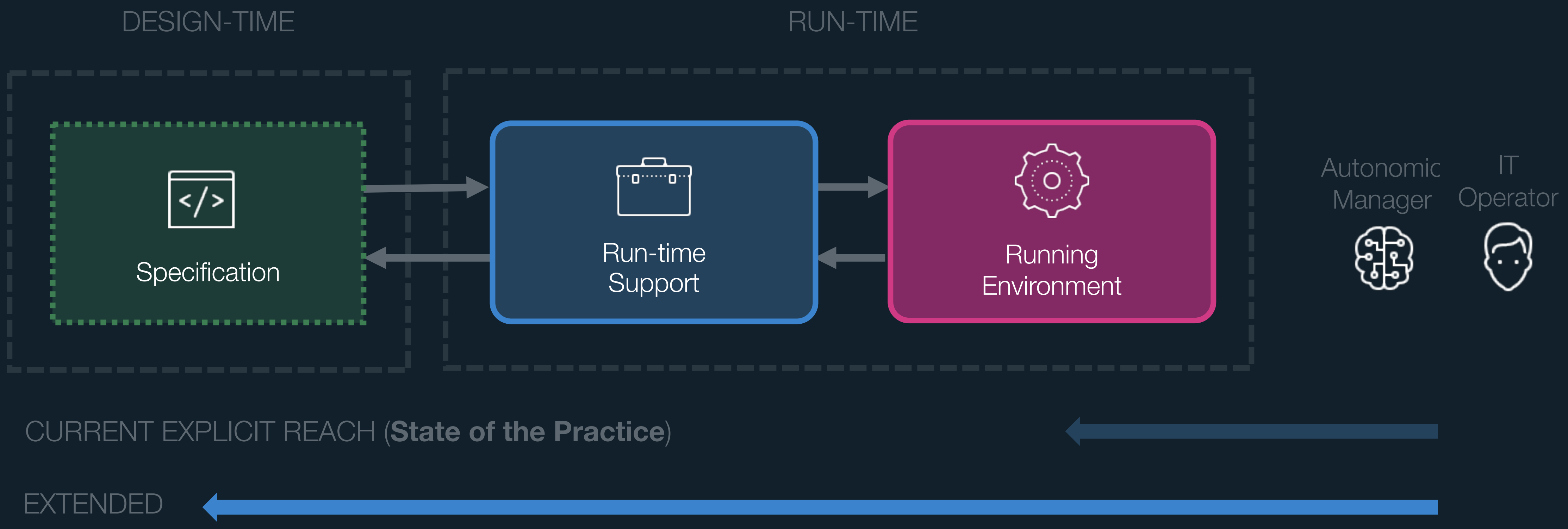
- Delay to reflect changes
- Extra time spent reviewing changes
- Merge conflicts are expected
- Specs and MART are inconsistent while pull request remains open unless the MART goes back to the previous version

Conflict Resolution

- Conflict resolution is not trivial
- Avoid formatting issues
- In case of conflicts, runtime components either:

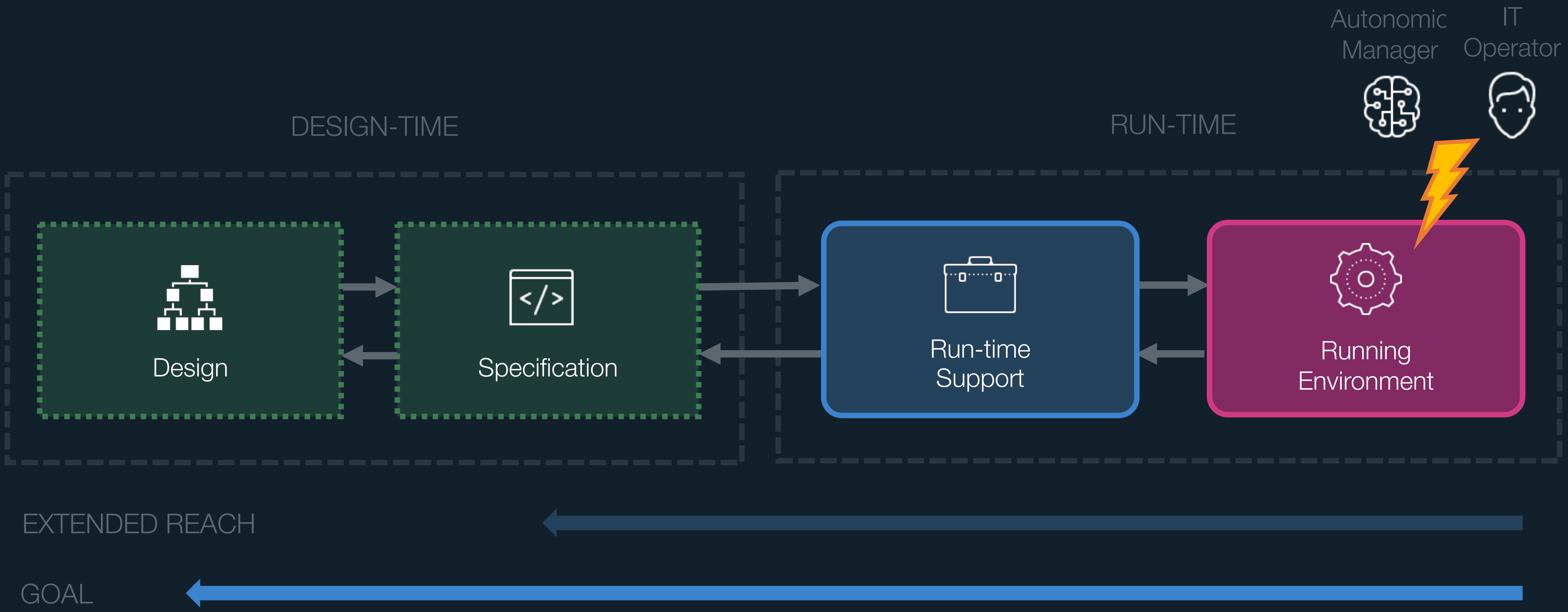
1. Drop the changes
2. Replace upstream changes

Future Work (1)

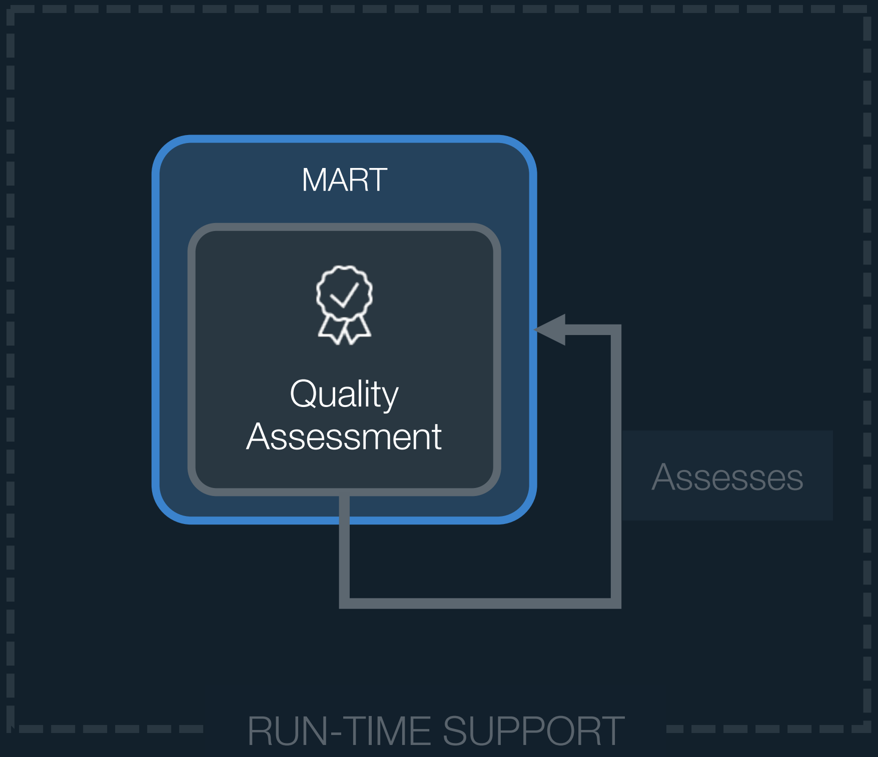


Future Work (1)

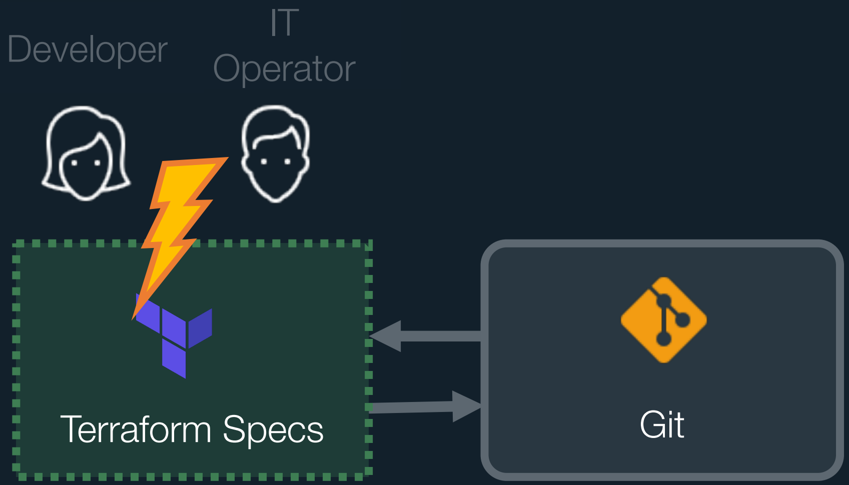
Design: the actual design artefacts or updated views of the running system (similar to database views)



Future Work (2)



1. Quality Assurance

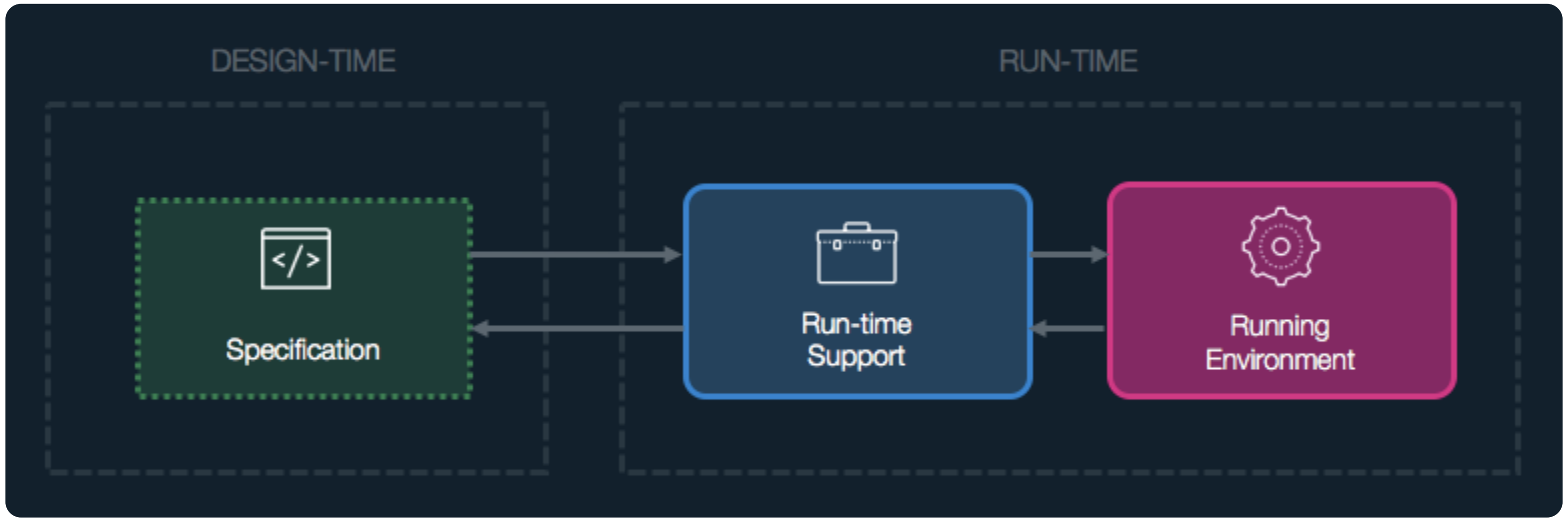


2. Tool Support

Conclusions

1. Problem:
Bi-directional Traceability in DevOps

2. Solution:
Two-way CI Framework



3. Future work:
Further Sync, Quality assurance, Tool support

Miguel Jiménez
miguel@uvic.ca